Package: quaqcr (via r-universe)

May 24, 2026

Title Quick ATAC-Seq QC

Version 1.0.4.9000

Description A wrapper around the 'quaqc' program described in Tremblay and Questa (2024) <doi:10.1093/bioinformatics/btae649>. 'quaqc' allows for assay for transposase-accessible chromatin using sequencing (ATAC-seq) specific quality control and read filtering of next-generation sequencing (NGS) data with minimal processing time and extremely low memory overhead. Any number of samples can be processed, using multiple threads if desired. 'quaqc' outputs a comprehensive set of aligned read metrics, including alignment size, fragment size, percent duplicates, mapq scores, read depth, GC content, and others. Although designed for ATAC-seq data, 'quaqc' can also be used for other unspliced DNA sequencing experiments (such as chromatin immunoprecipitation sequencing, or ChIP-seq) as many of the metrics are related to general sequencing quality. This R package also provides additional utilities for custom analyses and plotting of 'quaqc' results.

URL https://github.com/bjmt/quaqcr

BugReports https://github.com/bjmt/quaqcr/issues

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 3.6.0)

Imports methods, utils, jsonlite

Roxygen list(markdown = TRUE, old_usage = TRUE)

RoxygenNote 7.3.3

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

SystemRequirements quaqc (https://github.com/bjmt/quaqc)

Repository https://bjmt.r-universe.dev

Date/Publication 2026-05-24 09:50:20 UTC

RemoteUrl https://github.com/bjmt/quaqcr

RemoteRef HEAD

RemoteSha cb4a0ec40a7a378999bf97765c0358b38b65dc5c

Contents

`footprint` *Get transcription factor footprints with quaqc.*

Description

The TSS pileup feature of quaqc can instead be used to get single base resolution transcription factor footprints from ATAC-seq data. This function provides a convenient wrapper around such functionality. Target regions can be compared to unbound or background regions. (Note that no Tn5 bias correction is applied.)

Usage

```
footprint(target.motifs, bam.files, bkg.motifs = NULL, normalize = c("bkg",
  "rpm", "no"), tss.size = 501, tss.qlen = 1, tss.tn5 = TRUE,
  nfr = TRUE, verbose = 0, ...)
```

Arguments

`target.motifs`	Either (1) a filename of a BED file containing target motif positions, or (2) a `GRanges` object from the **GenomicRanges** package.
`bam.files`	Character vector of BAM file names. Must be coordinate sorted. If no index file can be found, `quaqc` will generate them.
`bkg.motifs`	Either (1) a filename of a BED file containing target motif positions, or (2) a `GRanges` object from the **GenomicRanges** package. (Optional.)
`normalize`	"bkg": Converts read density into values relative to the background (the first 25% of the window). "rpm": Conver to reads per million. "no": Return as the average number of reads per window.
`tss.size`	Integer, size of the TSS region for pileup.
`tss.qlen`	Integer, resize reads (centered on the 5-prime end for pileup.

tss.tn5	Logical, shift 5-prime end coordinates +4/-4 bases for pileup (was +4/-5 in quaqc <= 1.6). The shift can be customised via tn5.fwd and tn5.rev in quaqc >= 1.7.
nfr	Logical, turn on NFR mode.
verbose	Integer, a value from 0 to 2 for the level of program verbosity.
...	See quaqc().

Value

A data.frame containing read pileup data.

Author(s)

Benjamin Jean-Marie Tremblay, <benjmtremblay@gmail.com>

See Also

quaqc(), melt_reports()

Examples

```
bam <- "Sample.bam"
if (nzchar(Sys.which("quaqc")) && file.exists(bam)) {
  TATA_peaks <- system.file("extdata", "tata_p.bed.gz", package = "quaqcr")
  TATA_bkg <- system.file("extdata", "tata_n.bed.gz", package = "quaqcr")
  footprint(TATA_peaks, bam, bkg.motifs = TATA_bkg)
}
```

---

melt_reports *Melt sections of a quaqc report into a data.frame.*

Description

The quaqc report class type in R is divided into lists of lists, which can require additional manipulation. This function will "melt" these individual sections into data.frame objects.

Usage

```
melt_reports(report, section = c("bam_stats", "overview_unfilt",
  "overview_filt", "nucl_stats", "nucl_addn", "peak_stats", "tss_stats",
  "tss_pileup", "aln_hist", "frag_hist", "gc_hist", "depth_hist", "genome"),
  use.basename = FALSE, normalize.tss = c("no", "bkg", "rpm"),
  normalize.hist = c("no", "proportion", "max"))
```

Arguments

report	A quaqc object.
section	The quaqc object subsection to melt.
use.basename	Whether to use the base::basename() function on the sample names.
normalize.tss	How to normalize the TSS pileup. "no": Keep the signal as the average number of reads per window. "bkg": Calculate the signal relative to the background (the first 25% of the window). "rpm": Convert to reads per million.
normalize.hist	How to normalize the alignment size, fragment size, GC percent, and read depth histograms. "no": Keep as the total number of reads per bin. "proportion": Divide by the sum of reads across all windows. "max": Divide by the max bin count.

Value

A data.frame with varying columns based on the section being melted.

Author(s)

Benjamin Jean-Marie Tremblay, <benjmtremblay@gmail.com>

See Also

quaqc(), parse_quaqc_file()

Examples

```
report.file <- system.file("extdata", "report.json.gz", package = "quaqcr")
report <- parse_quaqc_file(report.file)
melt_reports(report, "overview_filt")
```

---

parse_quaqc *Parse a JSON quaqc report file.*

Description

Parse the output of quaqc --json into a easier to use quaqc-class object within R.

Usage

```
parse_quaqc(json.text)

parse_quaqc_file(json.file)
```

Arguments

`json.text`	The JSON report as a character vector.
`json.file`	The JSON report filename.

Details

A `quaqc` object is a higher level format encompassing the quaqc run parameters (accessible via `$metadata`) and the actual individual reports for each sample (accessible via `$reports`). The reports are themselves `quaqc_report`-class objects with multiple list slots, including:

- `$sample`: The filename of the sample.
- `$success`: Whether the sample was successfully analyzed.
- `$params`: Values for all quaqc parameters used to analyze this sample.
- `$genome`: Data about the genome taken from the BAM header.
- `$unfiltered`: Basic stats about the total number of reads before filtering.
- `$filtered`: Contains the majority of the data output by quaqc.

This final `$filtered` slot itself is broken down into several sub-lists:

- `$overview`: Average values for several stats such as fragment size.
- `$nuclear$stats`: Further breakdown of the previous stats for nuclear reads.
- `$nuclear$stats.warn`: Whether any quaqc parameters prevented it from accurately collecting some data.
- `$nuclear$addn.stats`: Genome coverage and the number of alignments without a MAPQ score.
- `$nuclear$histograms`: Raw histogram data for alignment size, fragment size, GC percent, and read depth.
- `$nuclear$peaks`: The number of peaks, the fraction of the effective genome covered by them, and the FRIP score.
- `$nuclear$tss`: The read pileup around TSSs as well as the TSS enrichment score.

Note that the word 'effective' refers to reads which are visible to quaqc within target regions or outside blacklisted regions, as well as reads associated with any specified target read groups.

Value

A `quaqc`-class object.

Author(s)

Benjamin Jean-Marie Tremblay, <benjmtremblay@gmail.com>

See Also

`quaqc()`

Examples

```
report.file <- system.file("extdata", "report.json.gz", package = "quaqcr")

## Option 1: parse a report already read into R
f <- gzfile(report.file, "rt")
json <- jsonlite::fromJSON(readLines(f), simplifyDataFrame = FALSE)
close(f)
report <- parse_quaqc(json)

## Option 2: parse a report directly from a file
report <- parse_quaqc_file(report.file)
```

---

pileup *Generate read pileups from BAMs with quaqc.*

Description

quaqc maintains an internal TSS pileup in order to calcualte a TSS enrichment score. This function takes advantage of this feature to instead generate read pileups for arbitrary sets of regions.

Usage

```
pileup(target.regions, bam.files, bkg.regions = NULL, normalize = c("rpm",
  "bkg", "no"), region.size = 5001, qlen = 0, verbose = 0, ...)
```

Arguments

`target.regions`	Either (1) a filename of a BED file containing target region positions, or (2) a `GRanges` object from the **GenomicRanges** package.
`bam.files`	Character vector of BAM file names. Must be coordinate sorted. If no index file can be found, quaqc will generate them.
`bkg.regions`	Either (1) a filename of a BED file containing target region positions, or (2) a `GRanges` object from the **GenomicRanges** package.
`normalize`	"bkg": Converts read density into values relative to the background (the first 25% of the window). "rpm": Conver to reads per million. "no": Return as the average number of reads per window.
`region.size`	The input regions will be uniformly resized to a single size.
`qlen`	The size of the reads when they are included in the pileup. A `qlen` of 0 means preserving the original read sizes; otherwise the reads are resized from their 5-prime ends.
`verbose`	Integer, a value from 0 to 2 for the level of program verbosity.
`...`	See `quaqc()`.

Value

A `data.frame` containing read pileup data.

Author(s)

Benjamin Jean-Marie Tremblay, <benjmtremblay@gmail.com>

See Also

`quaqc()`, `melt_reports()`

Examples

```
bam <- "Sample.bam"
if (nzchar(Sys.which("quaqc")) && file.exists(bam)) {
  peaks <- system.file("extdata", "peaks.bed.gz", package = "quaqcr")
  pileup(peaks, bam)
}
```

---

quaqc *Run quaqc from within R.*

Description

Interactive wrapper for running quaqc from within R. For a detailed description of the program, see the manual: execute `man quaqc` from the command line or open `doc/quaqc.1.md` in the program folder. For a brief description of the command parameters, as well as to see default values, call `quaqc()` without any arguments.

Usage

```
quaqc(bam.files, mitochondria = NULL, plastids = NULL, peaks = NULL,
  tss = NULL, target.names = NULL, target.list = NULL,
  blacklist = NULL, rg.names = NULL, rg.list = NULL, rg.tag = NULL,
  use.secondary = FALSE, use.nomate = FALSE, use.dups = FALSE,
  use.chimeric = FALSE, use.dovetails = FALSE, no.se = FALSE,
  mapq = NULL, min.qlen = NULL, min.flen = NULL, max.qlen = NULL,
  max.flen = NULL, use.all = FALSE, max.depth = NULL, max.qhist = NULL,
  max.fhist = NULL, tss.size = NULL, tss.qlen = NULL, tss.tn5 = FALSE,
  omit.gc = FALSE, omit.depth = FALSE, fast = FALSE, lenient = FALSE,
  strict = FALSE, nfr = FALSE, nbr = FALSE, footprint = FALSE,
  chip = FALSE, output.dir = NULL, output.ext = NULL, no.output = TRUE,
  json = "-", keep = FALSE, keep.dir = NULL, keep.ext = NULL,
  bedgraph = FALSE, bedgraph.qlen = NULL, bedgraph.tn5 = FALSE,
  bedgraph.dir = NULL, bedgraph.ext = NULL, bed = FALSE,
  bed.ins = FALSE, bed.tn5 = FALSE, bed.dir = NULL, bed.ext = NULL,
```

```
quant = NULL, quant.ins = FALSE, quant.tn5 = FALSE, quant.pn = FALSE,
call.peaks = FALSE, peaks.extsize = NULL, peaks.llocal = NULL,
peaks.qval = NULL, peaks.gsize = NULL, peaks.min.len = NULL,
peaks.max.gap = NULL, peaks.split = NULL, peaks.qscore = FALSE,
peaks.dir = NULL, peaks.ext = NULL, qscore.ext = NULL,
tn5.fwd = NULL, tn5.rev = NULL, threads = NULL, title = NULL,
continue = FALSE, verbose = 1, timeout = 0, env = character(),
stderr.file = "", bin = getOption("quaqc.bin"))
```

Arguments

bam.files	Character vector of BAM file names. Must be coordinate sorted. If no index file can be found, quaqc will generate them.
mitochondria	Character vector of mitochondria names. Provide "" to clear the defaults.
plastids	Character vector of plastid names. Provide "" to clear the defaults.
peaks	Either (1) a filename of a BED file containing peaks, or (2) a GRanges object from the **GenomicRanges** package.
tss	Either (1) a filename of a BED file containing TSSs, or (2) a GRanges object from the **GenomicRanges** package.
target.names	Character vector of sequence names to restrict quaqc.
target.list	Either (1) a filename of a BED file containing ranges to restrict quaqc, or (2) a GRanges object from the **GenomicRanges** package.
blacklist	Either (1) a filename of a BED file containing blacklist ranges, or (2) a GRanges object from the **GenomicRanges** package.
rg.names	A character vector of read group (RG) names to restrict quaqc.
rg.list	Filename of a text file containing read group (RG) names to restrict quaqc, one name per line.
rg.tag	Character, alternate SAM tag to match rg.names/rg.list against instead of the RG tag (e.g. "CB" for cell barcodes). Requires quaqc >= 1.3.
use.secondary	Logical, allow secondary alignments.
use.nomate	Logical, allow PE reads when the mate does not align properly.
use.dups	Logical, allow duplicate reads.
use.chimeric	Logical, allow supplemental or chimeric alignments.
use.dovetails	Logical, allow dovetailing PE reads.
no.se	Logical, discard SE reads.
mapq	Integer, min MAPQ score.
min.qlen	Integer, min alignment length.
min.flen	Integer, min fragment length.
max.qlen	Integer, max alignment length.
max.flen	Integer, max fragment length.
use.all	Logical, discard all filters and keep all reads.
max.depth	Integer, max base depth for read depth histogram.

max.qhist	Integer, max alignment length for histogram.
max.fhist	Integer, max fragment length for histogram.
tss.size	Integer, size of the TSS region for pileup.
tss.qlen	Integer, resize reads (centered on the 5-prime end for pileup.
tss.tn5	Logical, shift 5-prime end coordinates +4/-4 bases for pileup (was +4/-5 in quaqc <= 1.6). The shift can be customised via tn5.fwd and tn5.rev in quaqc >= 1.7.
omit.gc	Logical, omit calculation of read GC content.
omit.depth	Logical, omit calculation of read depths.
fast	Logical, turn on fast mode.
lenient	Logical, turn on lenient mode.
strict	Logical, turn on strict mode.
nfr	Logical, turn on NFR mode.
nbr	Logical, turn on NBR mode.
footprint	Logical, turn on footprinting mode.
chip	Logical, turn on ChIP-seq mode.
output.dir	Name of directory to save QC report if not that of input.
output.ext	Filename extension for output files.
no.output	Logical, suppress creation of output QC reports. Note that option is turned on by default when run from **quaqcr**.
json	Filename of JSON file to save combined QC results to. Set to NULL to suppress this. The default is to pipe the JSON output directly to R and not save to a file.
keep	Logical, save passing nuclear reads to a new BAM file.
keep.dir	Directory name to save filtered BAMs.
keep.ext	Extension of filtered BAMs.
bedgraph	Logical, output a gzipped read density bedGraph per sample. Requires quaqc >= 1.2. As of quaqc >= 1.7 the file is BGZF (block-gzip) and so is tabix-indexable, but still readable with gzip -dc/zcat.
bedgraph.qlen	Integer, resize reads (centered on the 5-prime end) for the bedGraph. Requires quaqc >= 1.2.
bedgraph.tn5	Logical, shift 5-prime end coordinates by the Tn5 offset (default +4/-4) for the bedGraph. Requires quaqc >= 1.2.
bedgraph.dir	Directory in which to write bedGraphs if not that of the input BAM.
bedgraph.ext	Filename extension for bedGraph output files.
bed	Logical, output a gzipped BED6 of passing reads per sample. Requires quaqc >= 1.4. As of quaqc >= 1.7 the file is BGZF, but still readable with gzip -dc/zcat.
bed.ins	Logical, write 5-prime insertion coordinates in BED3 instead of full-length alignments. Requires quaqc >= 1.4.
bed.tn5	Logical, adjust BED coordinates for the Tn5 shift. Requires quaqc >= 1.5.

bed.dir	Directory in which to write BED files if not that of the input BAM.
bed.ext	Filename extension for BED output files.
quant	File path to write a TSV of per-peak read counts (rows: peaks, columns: samples). Requires quaqc >= 1.5. The peaks themselves are supplied via peaks.
quant.ins	Logical, quantify based on 5-prime insertion coordinates instead of full alignments. Requires quaqc >= 1.5.
quant.tn5	Logical, adjust quantification coordinates for the Tn5 shift. Requires quaqc >= 1.5.
quant.pn	Logical, use pretty (basename) sample names in the quantification TSV header. Requires quaqc >= 1.5.
call.peaks	Logical, perform MACS3-style, no-control peak calling and write a per-sample narrowPeak.gz file. When peaks is not also supplied, the called peaks drive the FRIP value in the JSON report. Requires quaqc >= 1.7.
peaks.extsize	Integer, Tn5 insertion extension window for peak calling (default 150). Requires quaqc >= 1.7.
peaks.llocal	Integer, large local lambda window size for peak calling (default 10000). Requires quaqc >= 1.7.
peaks.qval	Numeric, q-value (FDR) cutoff for peak calling (default 0.05). Requires quaqc >= 1.7.
peaks.gsize	Numeric, effective genome size used for the local lambda and Benjamini-Hochberg correction (e.g. 1.2e8). Defaults to the effective nuclear genome size computed by quaqc. Requires quaqc >= 1.7.
peaks.min.len	Integer, minimum peak length (defaults to peaks.extsize). Requires quaqc >= 1.7.
peaks.max.gap	Integer, maximum gap between adjacent passing segments to merge into a single peak (default 1). Requires quaqc >= 1.7.
peaks.split	Numeric in (0, 1), enable splitting of multi-modal peaks at valleys whose pileup is less than peaks.split times the lower adjacent summit. Off by default. Requires quaqc >= 1.7.
peaks.qscore	Logical, also write a per-sample -log10(q) bedGraph track. Requires quaqc >= 1.7.
peaks.dir	Output directory for peak files.
peaks.ext	Filename extension for the narrowPeak output.
qscore.ext	Filename extension for the qscore bedGraph output.
tn5.fwd	Integer, override the global Tn5 shift for forward reads (default 4). Affects every *.tn5 option. Requires quaqc >= 1.7.
tn5.rev	Integer, override the global Tn5 shift for reverse reads (default 4 in quaqc >= 1.7; was effectively 5 in earlier versions). Requires quaqc >= 1.7.
threads	Integer, number of worker threads. Max one per sample.
title	Assign a title to run.
continue	Logical, do not return an error and instead continue running if samples trigger program errors.

verbose	Integer, a value from 0 to 2 for the level of program verbosity.
timeout	Integer, number of seconds before stopping quaqc. By default it is allowed to run indefinitely. See base::system2().
env	A character vector of environment variables to set when running quaqc. See base::system2().
stderr.file	Filename to save quaqc messages. By default they are printed in the console.
bin	Path to quaqc binary. Alternatively, set options(quaqc.bin). If the binary is present in the current working directory, provide "./quaqc". The default, set when **quaqcr** is loaded, is to assume the binary is present in your PATH.

Value

If nothing is provided to bam.files, then the help message is printed to the console and returned as a character vector, invisibly. Alternatively: if json = NULL then NULL, otherwise the JSON output as parsed by **jsonlite**.

Author(s)

Benjamin Jean-Marie Tremblay, <benjmtremblay@gmail.com>

See Also

base::system2(), parse_quaqc(), parse_quaqc_file()

Examples

```
if (nzchar(Sys.which("quaqc"))) {
  ## To check that you are properly linking to the binary and view help:
  quaqc()
}
```

---

quaqcr-pkg *quaqcr: Quick ATAC-seq QC in R*

Description

A wrapper around the 'quaqc' program alongside additional utilities.

Author(s)

Maintainer: Benjamin Jean-Marie Tremblay <benjmtremblay@gmail.com> (ORCID)

See Also

Useful links:

- https://github.com/bjmt/quaqcr
- Report bugs at https://github.com/bjmt/quaqcr/issues

`read_quaqc_output` *Read quaqc output files associated with a report.*

Description

These helpers locate the per-sample output files that quaqc produced for a given run and read them into `data.frames`. Output paths are reconstructed from each sample's BAM filename (stored in the report) combined with the directory and extension that quaqc would have used.

Usage

```
read_bedgraph(report, dir = NULL, ext = ".bedGraph.gz",
  reader = utils::read.table)

read_bed(report, dir = NULL, ext = ".bed.gz", reader = utils::read.table)

read_narrowpeak(report, dir = NULL, ext = ".narrowPeak.gz",
  reader = utils::read.table)

read_qscore(report, dir = NULL, ext = ".qscore.bedGraph.gz",
  reader = utils::read.table)

read_quant(report, file, reader = utils::read.table)
```

Arguments

`report`	A quaqc object (as returned by `quaqc()` or `parse_quaqc_file()`) or a single `quaqc_report` object.
`dir`	Optional directory containing the output files. If `NULL` (the default), the directory of each input BAM is used, matching the default behaviour of quaqc when no `*-dir` option is provided.
`ext`	Filename extension that quaqc appended to each sample's basename (after stripping the `.bam`/`.cram` suffix). The defaults match those of quaqc.
`reader`	Function used to read each file. The default `utils::read.table()` handles gzip and BGZF transparently. Replace with e.g. `data.table::fread` for large files.
`file`	Path to the quantification TSV produced by quaqc `--quant`. This file path is not stored in the JSON report, so it must be supplied explicitly.

Details

Column names are assigned based on each format:

- `read_bedgraph` / `read_qscore`: `chrom`, `start`, `end`, `value` (or `qscore`).
- `read_bed`: BED6 by default (`chrom`, `start`, `end`, `name`, `score`, `strand`), or BED3 (`chrom`, `start`, `end`) when `--bed-ins` was used. The format is detected from the boolean parameters in the report.
- `read_narrowpeak`: standard narrowPeak columns (`chrom`, `start`, `end`, `name`, `score`, `strand`, `signalValue`, `pValue`, `qValue`, `peak`).
- `read_quant`: read directly from the TSV header.

Reading the filtered BAM output (`--keep`) is not supported here; use a dedicated BAM-reading package such as **Rsamtools**.

Value

When `report` is a quaqc object, a named list of `data.frames` (one per successful sample, named by the BAM basename without the `.bam` suffix). When `report` is a single `quaqc_report`, a single `data.frame`. `read_quant()` always returns a single `data.frame`.

Author(s)

Benjamin Jean-Marie Tremblay, <benjmtremblay@gmail.com>

See Also

`quaqc()`, `parse_quaqc_file()`

Examples

```
bam <- "Sample.bam"
if (nzchar(Sys.which("quaqc")) && file.exists(bam)) {
  res <- quaqc(bam, bedgraph = TRUE, bed = TRUE, call.peaks = TRUE)
  bgs <- read_bedgraph(res)
  beds <- read_bed(res)
  peaks <- read_narrowpeak(res)
}
```

Index